# New Approximation Methods of Planar Offset and Convolution Curves

In-Kwon Lee[1], Myung-Soo Kim[1], and Gershon Elber[2]

[1] Dept. of Computer Science, POSTECH, Pohang 790–784, South Korea
[2] Dept. of Computer Science, Technion, IIT, Haifa 32000, Israel

**Summary.** We present new methods that approximate the offset and convolution of planar curves. They can be used as fundamental tools in various interesting geometric applications such as NC machining and collision detection of planar curved objects. Using quadratic curve approximation and tangent field matching, the offset and convolution curves can be approximated by polynomial or rational curves within the tolerance of approximation error $\epsilon > 0$. We suggest three methods of offset approximation that allow simple error analysis and at the same time provide high-precision approximation. Two methods of convolution approximation are also suggested that approximate convolution curves with polynomial or rational curves.

## 1. Introduction

Given two planar curved objects $O_1$ and $O_2$, their Minkowski sum $O_1 \oplus O_2$ is obtained by sweeping one object $O_2$ (with a fixed orientation) all over the other object $O_1$. Offset is a special case of Minkowski sum in which the sweeping object is restricted to be a circular disk.

Let $C_1(t)$ and $C_2(s)$ be the boundary curves of $O_1$ and $O_2$, respectively. Then each boundary point of $O_1 \oplus O_2$ is a point of the following form:

$$C_1(t) + C_2(s),$$

where $C_1'(t) \parallel C_2'(s)$ and $\langle C_1'(t), C_2'(s) \rangle > 0$, for some $s$ and $t$. We define the convolution curve $C_1 * C_2$ as follows:

$$(C_1 * C_2)(t) = C_1(t) + C_2(s(t)),$$

where $C_1'(t) \parallel C_2'(s(t))$ and $\langle C_1'(t), C_2'(s(t)) \rangle > 0$, for some reparameterization $s = s(t)$. That is, the convolution curve $C_1 * C_2$ is an envelope curve which is obtained by sweeping one curve $C_1$ (with a fixed orientation) along the other curve $C_2$ [3]. Offset curve is a special case of convolution curve in which the sweeping curve is restricted to be a circle.

The boundary $\partial(O_1 * O_2)$ is a subset of $(C_1 * C_2)(t)$, in general. Moreover, when both $O_1$ and $O_2$ are convex objects, the convolution curve $C_1 * C_2$ is exactly the same as $\partial(O_1 * O_2)$. Figure 1.1 shows the offset and convolution of planar convex curved objects. In Figure 1.1–(a), an offset is obtained by sweeping a circle along the boundary of an object and taking only the outer boundary of the envelope curve. In Figure 1.1–(b), a convolution is generated by sweeping one object along the boundary of the other and similarly taking the outer envelope curve. In Figure 1.1, the inner envelope curves do not contribute to the offset and convolution boundaries; their elimination is a direct consequence of the condition: $\langle C_1'(t), C_2'(s(t)) \rangle > 0$.

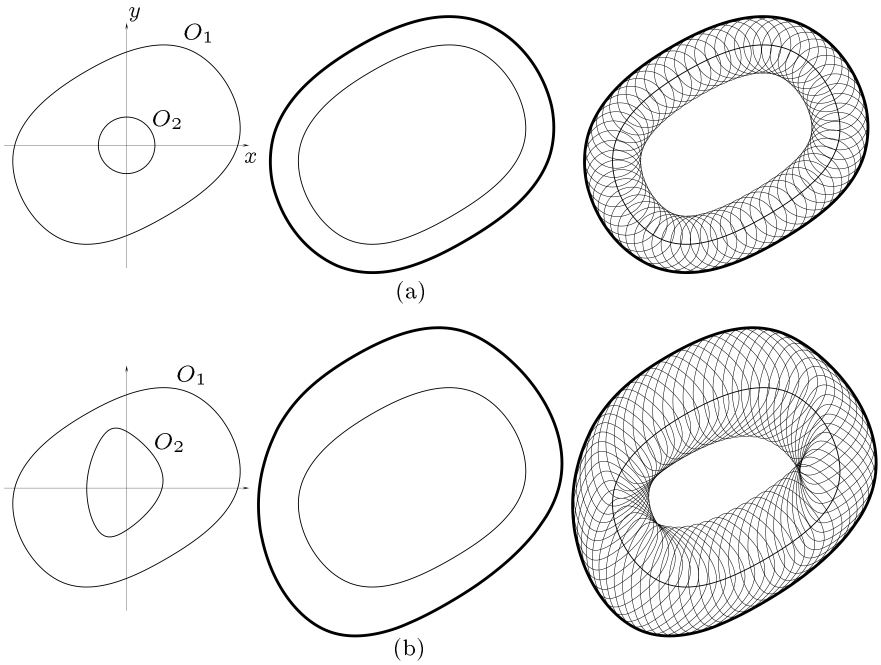



**Fig. 1.1.** (a) Offset and (b) convolution of planar objects

Offset and convolution computations are classic operations in CAD/CAM, which can be used in various interesting geometric applications such as NC machining [8, 18], motion planning [3, 17, 28], character font and brush stroke design [14, 15], blending [31], and shape transformation [22].

The exact offset and convolution curves of planar algebraic curves are algebraic, but not rational. Moreover, they have very high algebraic degree [12, 19, 23]. For example, the offset of a cubic Bézier curve has an algebraic degree of 10 [12]. Consequently, these fundamental limitations have led offset research to develop various offset approximation techniques [10, 23, 26]. Based on *quadratic curve approximation* and *tangent field matching*, we present new methods that approximate offset and convolution curves with polynomial or rational curves. Compared with other conventional approaches, our methods have many advantages: simple error analysis, high-precision approximation, and relatively small number of curve subdivisions.

This paper is organized as follows. Section 2 presents new algorithms to approximate planar offset curves. Using similar techniques, approximation algorithms for planar convolution curves are presented in Section 3. Finally, in Section 4, we conclude this paper and suggest further research problems.

# 2. Offset Approximation Methods

## 2.1 Planar Offset Curve

Let $C(t) = (x(t), y(t))$ be a planar rational curve. Then the offset curve $C_r(t)$ (with respect to a fixed radius $r$) is defined by:

$$C_r(t) = C(t) + r \cdot N(t), \tag{2.1}$$

where

$$N(t) = \frac{(y'(t), -x'(t))}{\sqrt{x'(t)^2 + y'(t)^2}} \tag{2.2}$$

is the unit normal of $C(t)$. Due to the square root function in the denominator of $N(t)$, the exact offset curve is not rational, in general [12].

Conventional offset curve approximation techniques are focused on approximating the exact offset curve, $C(t) + r \cdot N(t)$, with low degree spline curves. In this paper, we take different approaches which approximate and/or reparameterize the offset circle $r \cdot N(t)$ by planar polynomial or rational curves.

## 2.2 Previous Work

Elber et al. [10] reviewed the current state-of-the-art of offset approximation methods and classified conventional approaches according to the following two categories: (i) *control polygon based approach* [5, 7, 8, 32], and (ii) *interpolation based approach* [20, 21, 24, 30]. Control polygon based methods are usually much simpler. In these methods, it is relatively easy to construct an offset approximation curve so that it is completely inside/outside the exact offset curve. This capability of controlling offset under/over-estimation is quite useful in NC machining to prevent over-cut. Interpolation based methods usually generate more precise approximation, while generating much less curve data, especially for high-precision offset approximation.

Most of the previous methods estimate offset approximation error only at finite sample points, which may inaccurately reflect the global error. Moreover, each curve subdivision is made at the midpoint of the base curve, which may not be the optimal subdivision point. Elber and Cohen [8] estimated the offset approximation error by computing the maximum value of the difference function:

$$\psi(t) = \|C_r^a(t) - C(t)\|^2 - r^2, \tag{2.3}$$

where $C_r^a(t)$ is an approximation of $C_r(t)$. Curve refinement is made at the parameter value which gives the maximum error. This technique reduces the size of curve data.

Elber et al. [10] compared conventional offset approximation methods from quantitative as well as qualitative point of view. The difference function $\psi(t)$ of Equation (2.3) was used as a criteria to measure the performance of each method. This paper also uses the function $\psi(t)$ to evaluate the performance of each suggested method.

### 2.3 Offset Approximation Using Circle Approximation: CAO

Let $C(t) = (x(t), y(t))$, $t_0 \le t \le t_1$, be a planar regular parametric curve. Assume $C(t)$ has a one-to-one mapping $\theta_C$, called the *tangential angular map* of $C(t)$, defined by

$$\theta_C : t \in [t_0, t_1] \mapsto \theta_C(t) = \arctan \frac{y'(t)}{x'(t)} \in [0, 2\pi). \tag{2.4}$$

The unit normal vector field $N(t)$ has its curve trace as a unit circular arc.

To approximate the unit circular curve $N(t)$ (with a non-uniform speed) with a rational curve, we construct a quadratic curve: $Q(s) = (x_q(s), y_q(s))$, $s_0 \le s \le s_1$, which approximates the unit circular arc between two angles, $\arctan(y'(t_0)/x'(t_0))$ and $\arctan(y'(t_1)/x'(t_1))$. After that, using a suitable rational reparameterization $s(t)$, we can approximate $N(t)$ by a rational curve $Q(s(t))$. As a result, we can construct an approximated offset curve $C_r^a(t)$ as a rational curve:

$$C_r^a(t) = C(t) + r \cdot Q(s(t)), \tag{2.5}$$

where $s(t)$ is a proper reparameterization of $Q(s)$ satisfying the parallel relationship: $C'(t) \parallel Q'(s(t))$, $t_0 \le t \le t_1$. In the following discussion, we show how to construct a rational reparameterization $s(t)$.

The derivative curve of $Q(s)$ is a linear curve, $Q'(s) = (x_q'(s), y_q'(s)) = (as + b,\ cs + d)$, where $a$, $b$, $c$, and $d$ are some real constants. If $Q'(s(t)) = (as(t) + b, cs(t) + d)$ and $C'(t) = (x'(t), y'(t))$ have the same direction, we have

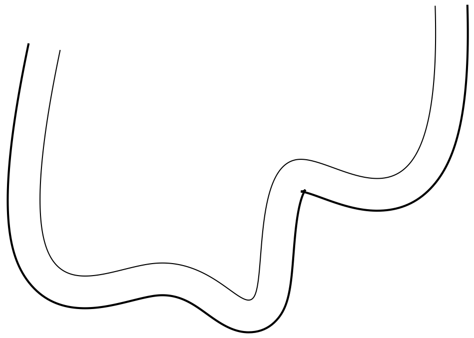
$$\frac{cs(t) + d}{as(t) + b} = \frac{y'(t)}{x'(t)}, \tag{2.6}$$

and the parameter $s(t) \in [s_0, s_1]$ is uniquely determined by

$$s(t) = \frac{by'(t) - dx'(t)}{cx'(t) - ay'(t)}. \tag{2.7}$$

Note that the solution $s(t)$ can be computed as a unique rational polynomial only when $Q(s)$ is a polynomial quadratic curve. The degree of the rational reparameterization $s(t)$ is $d-1$, where $d$ is the degree of $C(t)$. The composition $Q(s(t))$ and the offset approximation curve $C_r^a(t)$ can be computed by using symbolic computation tools for parametric curves [9, 11]. The degree of $C_r^a(t)$ is $3d-2$ for a polynomial curve $C(t)$, and $5d-4$ for a rational curve $C(t)$. This offset approximation method is called CAO (Circle Approximation Offset).

The offset approximation error is simply $r$ times that of approximating the unit circular arc by a quadratic curve $Q(s)$, assuming there is no local self-intersection loop in the computed offset approximation (see Reference [27] for more details). In spite of high degree in the resulting approximation, the size of output data from CAO is smaller than those of current offset approximation methods using the control polygon based approach. Figure 2.1 shows the CAO approximation of a cubic B–spline curve. The CAO approximation is a



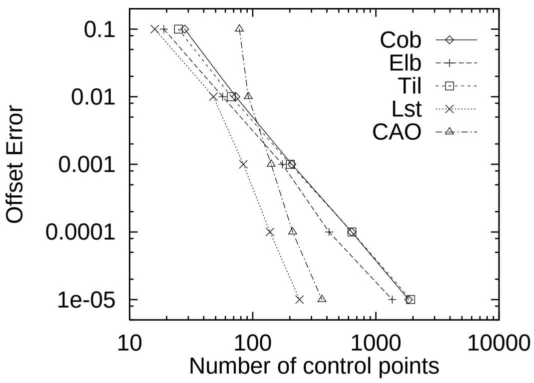



**Fig. 2.1.** CAO approximation (bold curve) of a cubic B–spline curve (light curve)

rational curve of degree 7. In Figure 2.1, the numbers of control points in offset approximation curves, generated from five different methods (*Cob* [5], *Elb* [8], *Til* [32], *Lst* [21], and CAO), are compared. The CAO method outperforms the control polygon based methods (*Cob* [5], *Elb* [8], and *Til* [32]), especially for high-precision approximation. (*Lst* [21] is an interpolation based method.)

The error in a circle approximation with quadratic Bézier curves can be estimated very quickly by evaluating a simple function [29]. Since the circle approximation error is the same as the error in CAO, we can estimate the offset approximation error in an *a priori* fashion (see Reference [27] for more details). That is, we can compute the error bound even without constructing an offset approximation curve. Figure 2.2 shows an approximation of the unit circle with three quadratic Bézier curve segments with $C^1$-continuity.

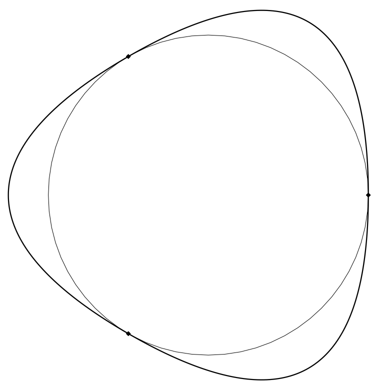


**Fig. 2.2.** Quadratic Bézier curve approximation of the unit circle

Data explosion is a serious problem in offset computation, especially for high-precision approximation. Therefore, we may store only a minimum amount of computation results such as curve subdivision parameters (in subdivision based methods [5, 7, 20, 24, 30, 32]), and generate all the other geometric information from these minimum data on the fly when they are required. In the CAO method, assuming rational curves of degree 7 are used for the offset curve approximation, this storage reduction scheme may reduce the output data size quite significantly (producing almost 14 times less data). Table 2.1 shows the result of this storage reduction technique applied to the examples shown in Figure 2.1. The numbers of base curve subdivisions are shown for subdivision based methods, as well as for an adapted version of least-squares based method [21] which exploits the subdivision based algorithm.

**Table 2.1.** Number of curve subdivisions in the offset approximation of Figure 2.1

| Tolerance | Cob | Elb | Til | Lst | CAO |
|---|---|---|---|---|---|
| $10^{-1}$ | 7 | 5 | 6 | 8 | 10 |
| $10^{-2}$ | 22 | 16 | 20 | 14 | 12 |
| $10^{-3}$ | 67 | 61 | 65 | 29 | 19 |
| $10^{-4}$ | 210 | 181 | 211 | 53 | 29 |
| $10^{-5}$ | 613 | 561 | 637 | 90 | 51 |

### 2.4 Offset Approximation by Tangent Field Matching: MO

The basic approach of CAO suggests another way of computing offset approximation curves. That is, instead of approximating the unit circule by polynomial quadratic curves, we can use an exact rational representation $Q(s)$ of the unit circle and approximate the reparameterization function $s(t)$ for the parallel relationship: $C'(t) \parallel Q'(s(t))$. This approach is based on the concept of tangent field matching which is defined as follows.

**Definition 2.1. Tangent Field Matching**
*Let $C_1(u)$, $u_0 \leq u \leq u_1$, and $C_2(v)$, $v_0 \leq v \leq v_1$, be two regular $C^1$ parametric curves. Assume two reparameterizations, $U : u \mapsto t$ and $V : v \mapsto t$, which map $C_1(u)$ into a curve $\hat{C}_1(t)$, $t_0 \leq t \leq t_1$, and $C_2(v)$ into a curve $\hat{C}_2(t)$, $t_0 \leq t \leq t_1$, respectively. If two unit tangent fields:*

$$T_1(t) = \frac{\hat{C}_1'(t)}{||\hat{C}_1'(t)||} \quad \text{and} \quad T_2(t) = \frac{\hat{C}_2'(t)}{||\hat{C}_2'(t)||}, \tag{2.8}$$

*are the same for all $t \in [t_0, t_1]$, the two curves $\hat{C}_1(t)$ and $\hat{C}_2(t)$ have a complete tangent field matching, and the two tangent fields of $C_1(u)$ and $C_2(v)$ are matched by the reparameterizations $U$ and $V$.* □

When the two curves have a complete tangent field matching, the inner product $\langle T_1(t), T_2(t) \rangle = 1$, for all $t \in [t_0, t_1]$. Cohen et al. [6] presented an algorithm to approximate the matching by solving the following optimization problem:

$$\max_{v(u)} \int_{u_0}^{u_1} \left\langle \frac{C_1'(u)}{||C_1'(u)||}, \frac{C_2'(v(u))}{||C_2'(v(u))||} \right\rangle du, \tag{2.9}$$

where $v(u) = V(U^{-1}(u))$, $v(u_0) = v_0$, and $v(u_1) = v_1$. This optimization problem is bounded from above by $u_1 - u_0$, since the normalized inner product does not exceed one. While the optimal solution of Equation (2.9) is difficult to obtain, we can solve instead an associated discrete optimization problem which may produce an arbitrarily close approximation to the solution of Equation (2.9). We sample both $C_1(u)$ and $C_2(v)$ at $n$ uniform parameter locations and compute their unit tangents as $T_{1,i}$ and $T_{2,j}$, $0 \leq i, j < n$. Then, the problem is reduced to a discrete optimization problem:

$$\max_{j(i)} \sum_{i=0}^{n-1} \left\langle T_{1,i}, T_{2,j(i)} \right\rangle, \tag{2.10}$$

subject to

$$j(0) = 0, \qquad j(n-1) = n-1, \qquad j(i) \leq j(i+1). \tag{2.11}$$

Cohen et al. [6] suggested a method to solve the optimization problem of Equation (2.10) within $O(n^2)$ time, where $n$ is the number of sample locations. This method employs a dynamic programming technique and provides a global optimal solution. The more sample points we use, the closer is the resulting reparametrized curve $C_2(v(u))$ to the completely matching curve with $C_1(u)$. Cohen et al. [6] also describe a method that approximates $v(u)$ from $j(i)$, by using a least-squares fitting with a B–spline curve. The composition of $C_2(v(u))$ can then be computed using symbolic computation tools [9, 11], while resulting in a B–spline curve representation of the offset approximation.

The offset approximation method: $C_r^a(t) = C(t) + r \cdot Q(s(t))$, where $s(t)$ is approximated by tangent field matching, is called MO(Matching Offset). One difficulty in MO is that we cannot estimate the approximation error by using the difference function $\psi(t)$ of Equation (2.3). In MO, the function $\psi(t)$ is useless since it always vanishes:

$$\psi(t) = \|C_r^a(t) - C(t)\|^2 - r^2 = \|r \cdot Q(s(t))\|^2 - r^2 = 0. \tag{2.12}$$

Because only $s(t)$ is approximated, the error in $s(t)$ results in a vector $r \cdot Q(s(t))$ which may be different from the normal direction $N(t)$ of $C(t)$. While the distance between the original curve and the offset approximation curve, for any parameter value $t$, remains $r$ throughout, the error will be introduced in the form of the associated angular deviation. The curve tangent $C'(t)$ is expected to be perpendicular to $Q(s(t))$. However, since we are using $s(t)$ which is only an approximation, the orthogonality condition is not completely satisfied. Computing the deviation from the parallel condition: $N(t) \parallel Q(s(t))$, is one way to measure the error in the offset approximation of MO.

The error functional

$$\delta(t) = \frac{\langle C'(t), Q(s(t))\rangle^2}{||C'(t)||^2}, \tag{2.13}$$

computes the value of $\cos^2\alpha$, where $\alpha$ is the angle between $C'(t)$ and $Q(s(t))$ (see Figure 2.3). (Note that the normal of $Q(s)$ is the same as $Q(s)$.) This angle, $\alpha$, must be $\frac{\pi}{2}$ for an exact offset curve. When $\theta$ denotes the angle $||\alpha - \frac{\pi}{2}||$, the distance between the exact offset point $C_r(t)$ and an approximated offset point $C_r^a(t)$ is $\epsilon = 2r\sin\frac{\theta}{2} \approx r\sin\theta = r\cos\alpha$.

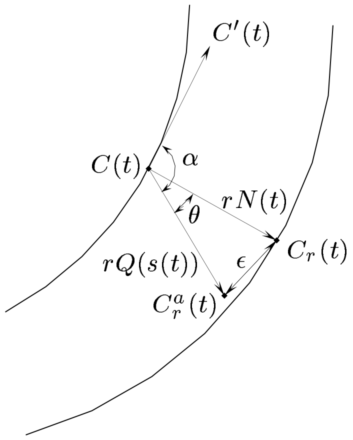



**Fig. 2.3.** Measuring the error in the offset approximation by tangent field matching

The resulting offset approximation curve is also rational. However, the degree is much lower, i.e., $d + 2$ (when we use a linear reparameterization $s(t)$), both for rational as well as polynomial base curves $C(t)$. Figure 2.4 shows an example of MO (Figure 2.4–(a)), along with its error functional $\delta(t)$ (Figure 2.4–(b)). In Figure 2.5, we show MOs of a cubic Bézier curve, $C(t)$, $t_0 \leq t \leq t_1$, for different tolerances. Examine the line segments which connect $C(t_i)$'s and the corresponding $C_r^a(t_i)$'s, where $t_0 \leq t_i \leq t_1$. We can realize that the parallel relationship: $C'(t)||C_r'^a(t)$, is enhanced as the precision of tolerance is increased.

## 2.5 Combined Approach: CAMO

One disadvantage of MO and CAO is that they generate a rational offset curve even for a polynomial base curve. In the case of MO, the exact circle $Q(s)$ in the offset approximation $C(t) + r \cdot Q(s(t))$ is inherently a rational quadratic curve. On the other hand, in CAO, the reparameterization $s(t)$ is a rational polynomial, while the curve $Q(s)$ is a polynomial quadratic curve.

This observation naturally suggests a new approach which adapts these two methods. That is, we use a quadratic polynomial approximation for $Q(s)$, and also use the tangent field matching technique to approximate $s(t)$ with

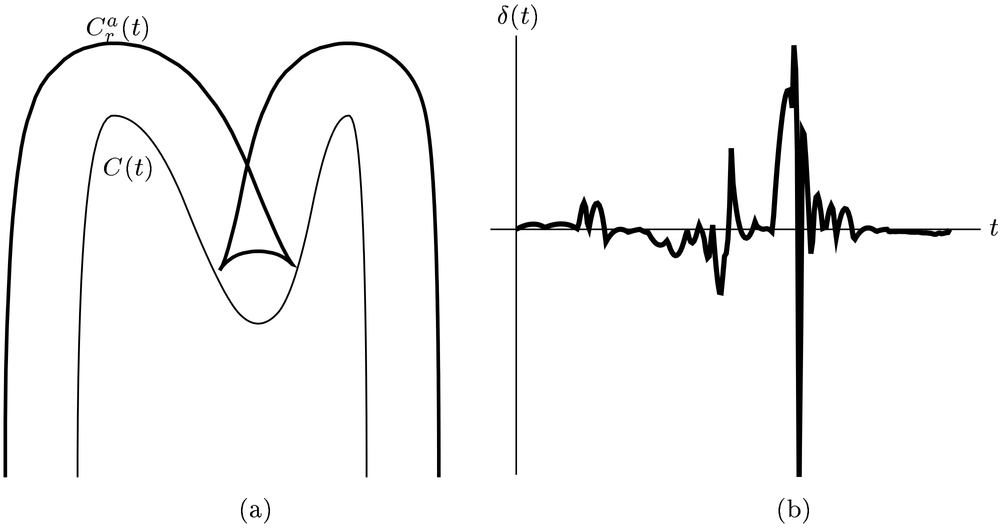



**Fig. 2.4.** MO and $\delta(t)$

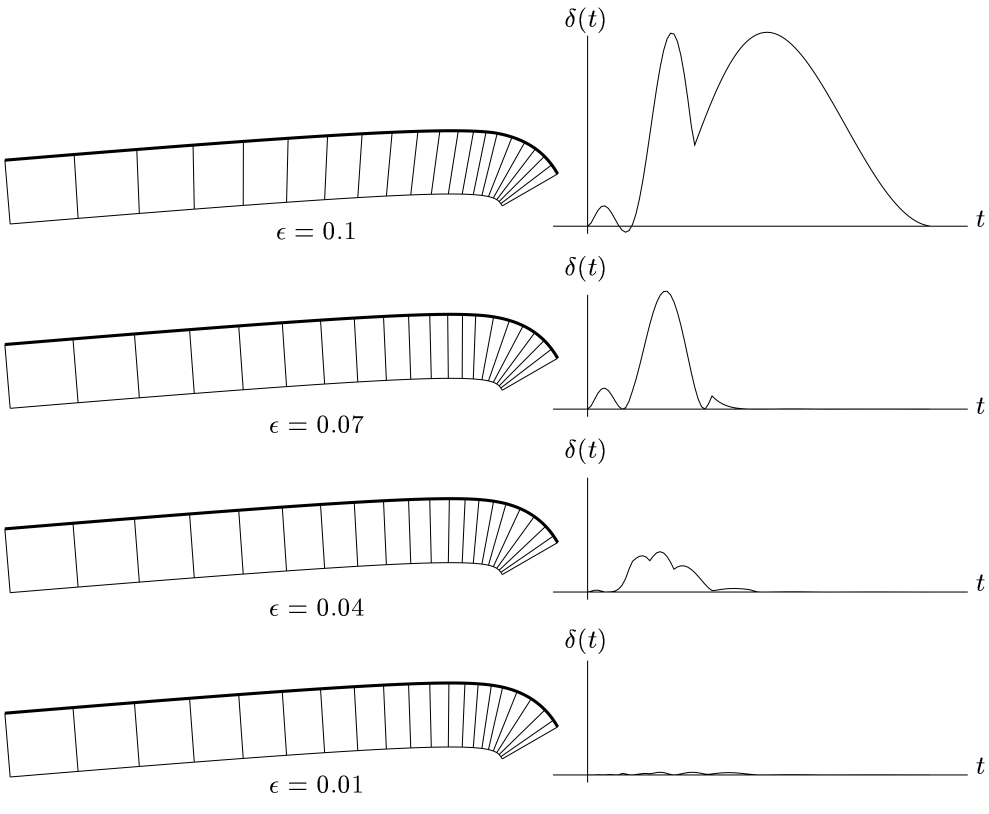



**Fig. 2.5.** MOs (bold curves) for a cubic Bézier curve for Different Tolerances.

a polynomial. Then, the resulting offset approximation curve $C_r^a(t)$ becomes a polynomial curve with the same degree as the base curve, when the base curve is a polynomial curve and the reparameterization polynomial $s(t)$ is linear. When we apply this method to a rational base curve, the degree of $C_r^a(t)$ is $d+2$, where $d$ is the degree of the base curve. We call this offset approximation method CAMO (Circle Approximation Matching Offset).

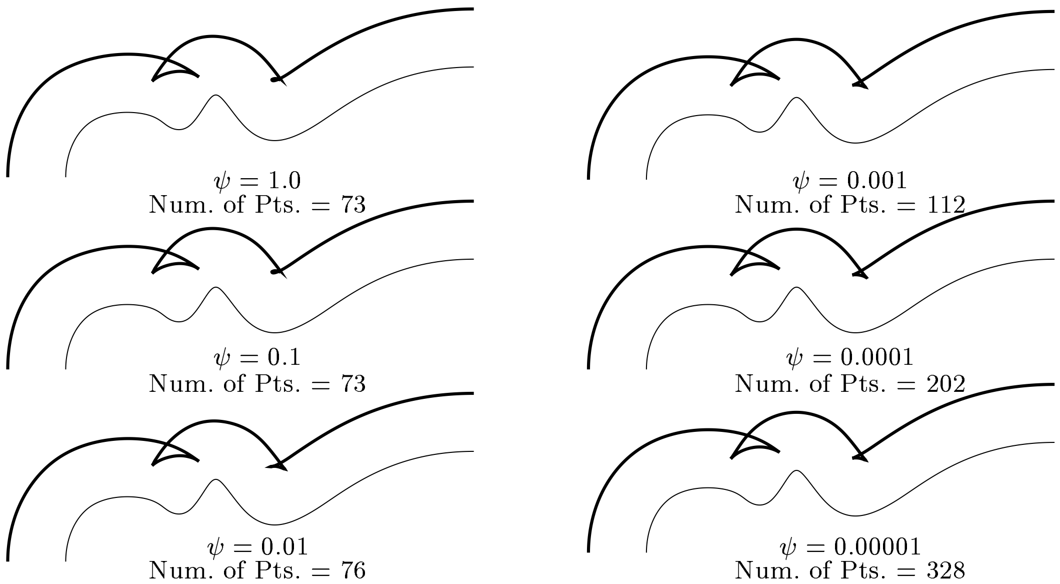



**Fig. 2.6.** Polynomial CAMOs (bold curves) of a cubic polynomial B–spline curve.

The method CAMO cannot use the error functional $\delta(t)$ of Equation (2.13). This is because $Q(s(t))$ is not an exact unit circular arc. The modified error functional $\delta(t)$ is defined as follows:

$$\delta(t) = \frac{\langle C'(t), Q(s(t)) \rangle^2}{||C'(t)||^2 ||Q(s(t))||^2}. \tag{2.14}$$

In CAMO, we also need to consider the difference function $\psi(t)$ of Equation (2.3), because CAMO does not use an exact rational quadratic curve to represent a unit circle. That is, there is no guarantee for the condition of Equation (2.12). Thus, CAMO uses two tolerances of approximation error, $\psi$ for the difference function $\psi(t)$, and $\epsilon$ for the function $\delta(t)$ of Equation (2.14).

Figure 2.6 shows some CAMO curves of a cubic polynomial B–spline base curve for different tolerances of $\psi$ and a fixed $\epsilon$. All the offset approximation curves are cubic polynomial B–spline curves.

# 3. Convolution Approximation Methods

## 3.1 Planar Convolution Curve

Convolution is a classic operation which has been used as a tool for computing collision–free paths in robot motion planning [3, 17, 28]. Moreover, the convolution operation has applications in character font design [14, 15], offset and rounding [31], and shape transformation [22].

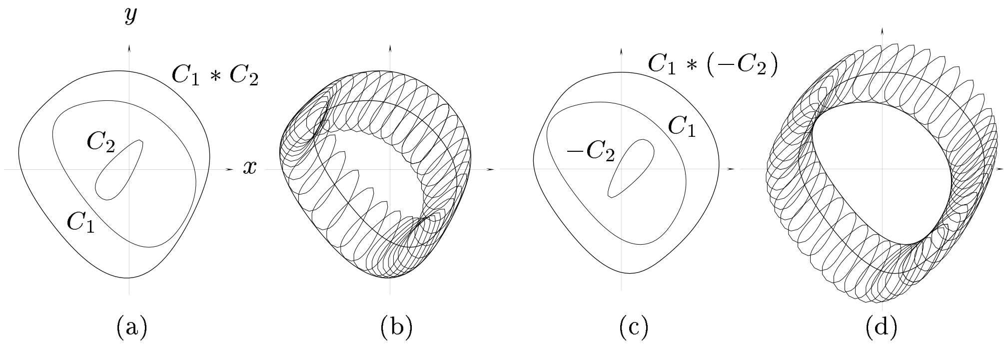



**Fig. 3.1.** Convolution and *C–space* obstacle

Given two regular parametric curves: $C_1(t) = (x_1(t), y_1(t)),\ t_0 \leq t \leq t_1$, and $C_2(s) = (x_2(s), y_2(s)),\ s_0 \leq s \leq s_1$, we define their convolution curve $(C_1 * C_2)(t)$ as follows (see [3, 17]):

$$(C_1 * C_2)(t) = C_1(t) + C_2(s(t)), \tag{3.1}$$

where $C_1'(t) \parallel C_2'(s(t))$ and $\langle C_1'(t), C_2'(s(t))\rangle > 0$, for some reparameterization $s = s(t)$. Figure 3.1–(a) shows the convolution curve $C_1 * C_2$ of two closed convex curves $C_1$ and $C_2$. The convolution curve $C_1 * C_2$ is an envelope curve which is obtained by sweeping one curve $C_1$ (with a fixed orientation) along the other curve $C_2$ [3] (see Figure 3.1–(b)).

Convolution curve can be used as a tool for collision detection between two planar curved objects. Let $O_1$ and $O_2$ denote planar curved objects bounded by closed curves $C_1(t)$ and $C_2(s)$, respectively. Their *Minkowski sum* $O_1 \oplus O_2$ is defined by

$$O_1 \oplus O_2 = \{a + b \mid a \in O_1,\ b \in O_2\}. \tag{3.2}$$

The boundary $\partial(O_1 \oplus O_2)$ of the Minkowski sum is a subset of the convolution curve $C_1 * C_2$ [3, 14, 23]. When both $O_1$ and $O_2$ are convex objects, the convolution curve $C_1 * C_2$ is exactly the same as $\partial(O_1 \oplus O_2)$.

In Figure 3.1–(c), we compute the convolution $C_1 * (-C_2)$, where $-C_2(s) = (-x_2(s), -y_2(s))$ is the symmetric curve of $C_2$ with respect to the local reference point (which is located at the origin). Similarly, we define $-O_2 = \{-b \mid b \in O_2\}$ as the reversed object of $O_2$ with respect to the origin. There is no collision between $O_1$ and $O_2$ as long as the reference point of $O_2$ does not penetrate into the Minkowski sum $O_1 \oplus (-O_2)$ which is bounded by the convolution curve $C_1 * (-C_2)$. In Figure 3.1–(d), the moving object $O_2$ has continous contact with the obstacle $O_1$ as it moves (with a fixed orientation) while its reference point follows along the convolution curve $C_1 * (-C_2)$. The object $O_1 * (-O_2)$ is called the *Configuration–space (C–space) obstacle* of the obstacle $O_1$ with respect to the moving object $O_2$. The *C–space* obstacle has been used as a tool for collision-detection in robot motion planning [3, 25, 28].

### 3.2 Previous works

Ghosh [15, 16] presented the concepts, algorithms, and data structures for various Minkowski operations in great detail. Guibas et al. [17] investigated some important properties of convolution curves. Convolution has a close relationship to other geometric entities such as common tangent and curve intersection. Lozano-Pérez [28] used the convolution technique to compute the *C–space* obstacles of polygonal objects.

Bajaj and Kim [3] discussed important issues in *C-space* generation: convolution curve generation, compatible subdivision of input curves, and the elimination of redundant convolution curve segments. They assumed exact algebraic representation of convolution curves which can be obtained by solving simultaneous algebraic equations, where input curves may be implicit and/or parametric. However, the resulting algebraic degrees are too high to be useful in practice. Ghosh [14, 16] demonstrated that Equation (3.5) has a closed-form solution in some special cases. Most of his results are restricted to closed convex curves such as ellipse; moreover, the solutions are not polynomial or rational curves. Kohler and Spreng [25] suggested a method to compute the exact convolution curve in a parametric form, while providing some numerical techniques to speed up the computation; however, the parametric curves are not polynomial or rational curves, in general.

Lee and Kim [26] suggested an approximation method to compute the convolution curve. In a preprocessing step, the input curves $C_1$ and $C_2$ are approximated by a sequence of discrete points $C_1(t_j)$ and $C_2(s_j)$ $(j = 0, \ldots, k)$ at which they have predefined normal directions $n_j$. A simple vector addition $C_1(t_j) + C_2(s_j)$ generates a sequence of discrete points on $C_1 * C_2$ and thus constructs a piecewise linear approximation to the convolution curve $C_1 * C_2$. Kaul and Farouki [23] suggested another piecewise linear approximation. Without using any preprocessing, they generated a sequence of discrete points along the convolution curve on the fly, by computing Equation (3.5) for two input curve segments. Ahn et al. [1] considered the general sweep in which the moving object may change its shape and orientation dynamically while tracing along a trajectory curve. They computed the boundary of a general sweep by approximating the sweep envelope curve with line segments.

As in the case of offset computation, it is desirable to develop efficient approximation methods for convolution curves. The previous methods either represent the convolution curve exactly by a non-rational algebraic curve or construct a piecewise linear approximation to the convolution curve. Linear approximations may not be acceptable to conventional CAD systems due to data explosion. In this paper, using the techniques of quadratic approximation and tangent field matching, we suggest new methods to approximate convolution curves by polynomial or rational curves.

### 3.3 Compatible subdivision

Let $C_1(t) = (x_1(t), y_1(t))$, $t_0 \le t \le t_1$, and $C_2(s) = (x_2(s), y_2(s))$, $s_0 \le s \le s_1$, be two regular parametric curves. Assume there is a one-to-one correspondence between $t \in [t_0, t_1]$ and $s \in [s_0, s_1]$, while satisfying the condition: $C_1'(t) \parallel C_2'(s)$ and $\langle C_1'(t), C_2'(s)\rangle > 0$; that is, $N_1(t) = N_2(s)$, where $N_1(t)$ and $N_2(s)$ denote the unit outward normal vectors (see Equation (2.2)) of $C_1(t)$ and $C_2(s)$. Then the two curves $C_1(t)$ and $C_2(s)$ are called *compatible*.

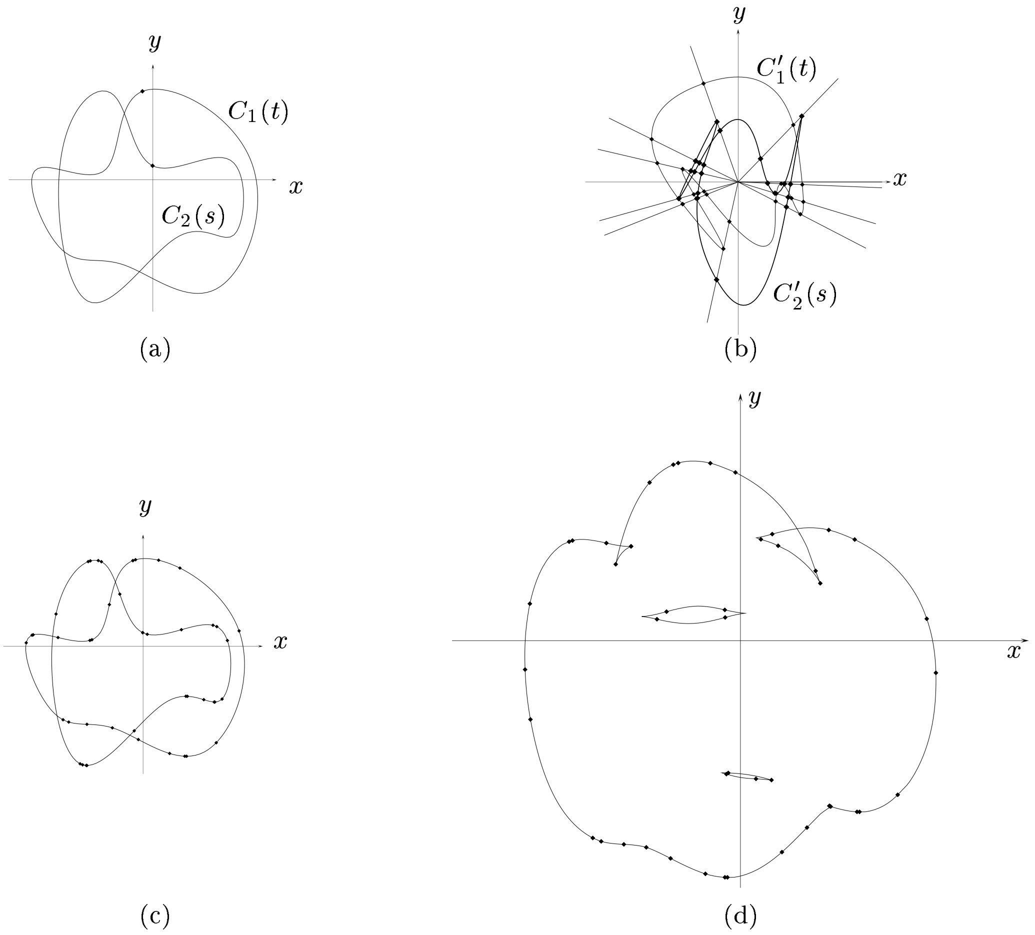



**Fig. 3.2.** Hodograph subdivision

When two input curves $C_1(t)$ and $C_2(s)$ are closed curves, but not convex, they are not compatible each other. In this case, for the generation of convolution curves, we need to subdivide the input curves into compatible subsegments. Figure 3.2 shows a simple method (called *hodograph subdivision*) which subdivides $C_1(t)$ and $C_2(s)$ into compatible subsegments. Let $C_1(t_i)$ and $C_2(s_j)$ ($i = 1, \ldots, m_0$, and $j = 1, \ldots, n_0$) be all the inflection points of $C_1(t)$ and $C_2(s)$, respectively. The inflection points of a parametric curve, $C(u) = (x(u), y(u))$, can be computed by solving the following equation:

$$x'(u)y''(u) - x''(u)y'(u) = 0, \tag{3.3}$$

using symbolic and numeric computation tools [9]. Assume $C(u_0)$ is an inflection point. Then, in the hodograph $C'(u)$, the ray $\overline{OC'(u_0)}$ emanating from the origin $O$ and passing through the point $C'(u_0)$ is tangent to the hodograph $C'(u)$. The set of rays:

$$\mathcal{L} = \{ \overline{OC_1'(t_i)}, \overline{OC_2'(s_j)}, \overline{OC_1'(t_0)}, \overline{OC_2'(s_0)} \mid i = 1, \ldots, m_0, \text{ and } j = 1, \ldots, n_0 \} \tag{3.4}$$

subdivides the plane $R^2$ into sector-form regions $R_k$ ($k = 1, 2, 3, ..., l$); see Figure 3.2–(b).

After computing all the inflection points, the hodographs $C_1'(t)$ and $C_2'(s)$ are subdivided into piecewise curves $C_{1,i}'(t)$ ($i = 1, 2, 3, ..., m$) and $C_{2,j}'(s)$ ($j = 1, 2, 3, ..., n$) at all the intersection points with each ray in the set $\mathcal{L}$ of Equation (3.4). These intersection points can also be computed using symbolic and numeric computation tools [9]. The original input curves, $C_1(t)$ and $C_2(s)$, can be subdivided into subsegments, $C_{1,i}(t)$ and $C_{2,j}(s)$ ($i = 1, 2, 3, ..., m$ and $j = 1, 2, 3, ..., n$), at the same subdivision parameters for $C_{1,i}'(t)$ and $C_{2,j}'(s)$, respectively (see Figure 3.2–(c)).

For a subsegment $C_{1,i}(t)$, let $R_k$ be the sector-form region which contains the hodograph subsegments $C_{1,i}'(t)$. Then, for each $C_{2,j}'(s)$ contained in $R_k$, the subsegments $C_{1,i}(t)$ and $C_{2,j}(s)$ are compatible each other. Figure 3.2–(d) shows a graph of convolution curve segments, in which each convolution curve segment is generated for each compatible pair of $C_{1,i}(t)$ and $C_{2,j}(s)$. In the following two subsections, we discuss how to approximate each convolution curve segment by a polynomial or rational curve.

### 3.4 Convolution Approximation Using Quadratic Curve Approximation: QAC

Let $C_1(t) = (x_1(t), y_1(t))$, $t_0 \le t \le t_1$, and $C_2(s) = (x_2(s), y_2(s))$, $s_0 \le s \le s_1$, be two compatible curve segments. The parallel relationship, $C_1'(t) \parallel C_2'(s)$, requires one to establish a proper reparameterization of $s = s(t)$ as a function of $t$. The equation representing the parallel relationship is given by:

$$y_2'(s)x_1'(t) - x_2'(s)y_1'(t) = 0. \tag{3.5}$$

Even if $C_1(t)$ and $C_2(s)$ are compatible, we do not have a unique algebraic representation of $s(t)$, because Equation (3.5) may not have a unique closed-form solution for the curves of arbitrary degree.

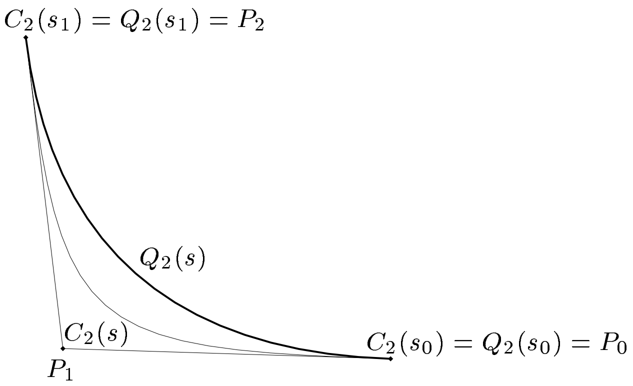



**Fig. 3.3.** Approximation of $C_2(s)$ by quadratic Bézier curve $Q_2(s)$ (bold curve)

When we approximate $C_2(s)$ by a quadratic curve, $Q(s) = (x_q(s), y_q(s))$, as in Figure 3.3, we can represent $s(t)$ as a rational function of $t$. The approximated convolution curve $C_1(t) *^a C_2(s)$ can be represented as follows:

$$C_1(t) *^a C_2(s) = C_1(t) + Q_2(s(t)), \quad t_0 \le t \le t_1, \tag{3.6}$$

where $s(t)$ is the reparameterization satisfying the following equation:

$$y'_q(s)x'_1(t) - x'_q(s)y'_1(t) = 0. \tag{3.7}$$

The reparameterization function $s(t)$ can be computed by using a formula similar to Equation (2.7), which results in a unique parametric representation of $C_1(t) + Q_2(s(t))$. When the approximation error between $Q_2(s)$ and $C_2(s)$ is $\epsilon$, the resulting approximated convolution curve $C_1(t) + Q_2(s(t))$ also has the same approximation error $\epsilon$. (In fact, this is true only for non-trimmed (closed or infinite) input curves and for the convolution curves after eliminating self–intersection loops; see Reference [27] for more details.) We call this method QAC (Quadratic Approximation Convolution). QAC is similar to the offset approximation method CAO in Section 2.3 which is based on the approximation of a circular arc with quadratic curves.

### 3.5 Convolution Approximation Using Tangent Field Matching: TMC

The other method directly approximates the reparameterization $s(t)$, rather than approximating the curve $C_2(s)$ using tangent field matching which was introduced in Section 2.4.

The tangent field matching allows the computation of approximated convolution between $C_1(t)$ and $C_2(s)$, by computing a proper reparameterization $s(t)$ for $C_2(s(t)) = (x_2(s(t)), y_2(s(t)))$. Unfortunately, the approximation error cannot be computed with the distance function $\psi(t)$ which we use for the quadratic curve approximation. Instead of the distance function $\psi(t)$, we use the following formula which represents the value of $\cos^2 \alpha$, where $\alpha$ is the angle between $C_1'(t)$ and $N_2(s(t))$:

$$\delta(t) = \frac{\langle C_1'(t), N_2(s(t))\rangle^2}{||C_1'(t)||^2}, \tag{3.8}$$

where

$$N_2(s(t)) = \frac{(y_2'(s(t)), -x_2'(s(t)))}{\sqrt{x_2'(s(t))^2 + y_2'(s(t))^2}}$$

is the unit normal vector field of $C_2(s(t))$. Then, the angle deviation can be represented as $\epsilon = \left\| \frac{\pi}{2} - \arccos(\sqrt{\max \delta(t)}) \right\|$. We call this method TMC (Tangent field Matching Convolution).

### 3.6 Comparisons and Examples

Each of the two approximation methods, QAC and TMC, has its own relative advantages and disadvantages. QAC is much simpler to compute than TMC. Moreover, we can guarantee that the resulting approximated convolution curve is within $\epsilon$-distance from the exact convolution curve, where $\epsilon$ is the tolerance of quadratic curve approximation. When both $C_1(t)$ and $C_2(s)$

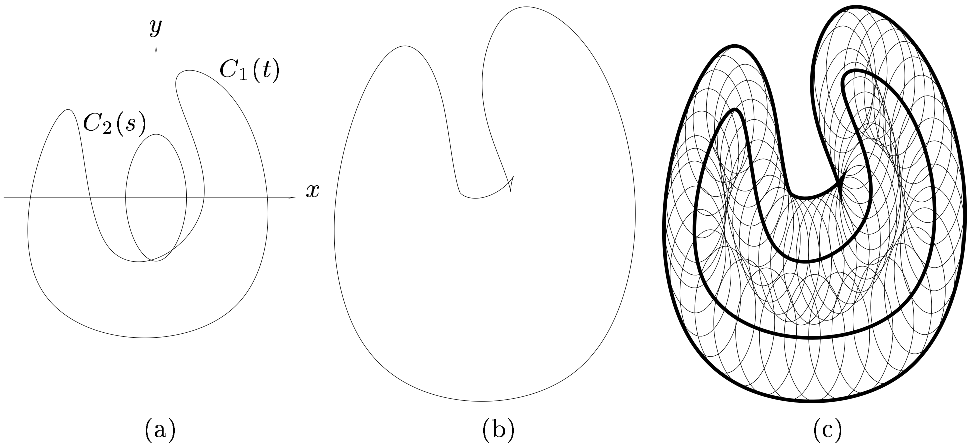
**Table 3.1.** Degrees of QAC and TMC approximations

| $C_1$ | $C_2$ | QAC | TMC |
|---|---|---|---|
| $d_1$ | $d_2$ | rational $3d_1 - 2$ | $\max(d_1, d_2)$ |
| $d_1$ | rational $d_2$ | rational $3d_1 - 2$ | rational $d_1 + d_2$ |
| rational $d_1$ | $d_2$ | rational $5d_1 - 4$ | rational $d_1 + d_2$ |
| rational $d_1$ | rational $d_2$ | rational $5d_1 - 4$ | rational $d_1 + d_2$ |

are polynomial curves, TMC generates lower degree approximation curves than QAC (see Table 3.1). The degree of QAC approximation depends only on the degree of $C_1(t)$.

When we approximate $C_1(t)$ by another quadratic curve $\bar{Q}_1(t)$, the QAC approximation (applied to $\bar{Q}_1(t)$ instead of $C_1(t)$) is a rational curve of degree four. Figure 3.4 shows the degree and the number of control points of various QAC approximations computed from four different combinations of input curves. The QAC approximation shown in Figure 3.4–(b) is computed from a cubic B–spline curve $C_1(t)$ and a rational cubic B–spline curve $C_2(s)$.

Figure 3.4–(c) verifies the convolution curve by sweeping $C_2(s)$ (family of light curves) along $C_1(t)$.



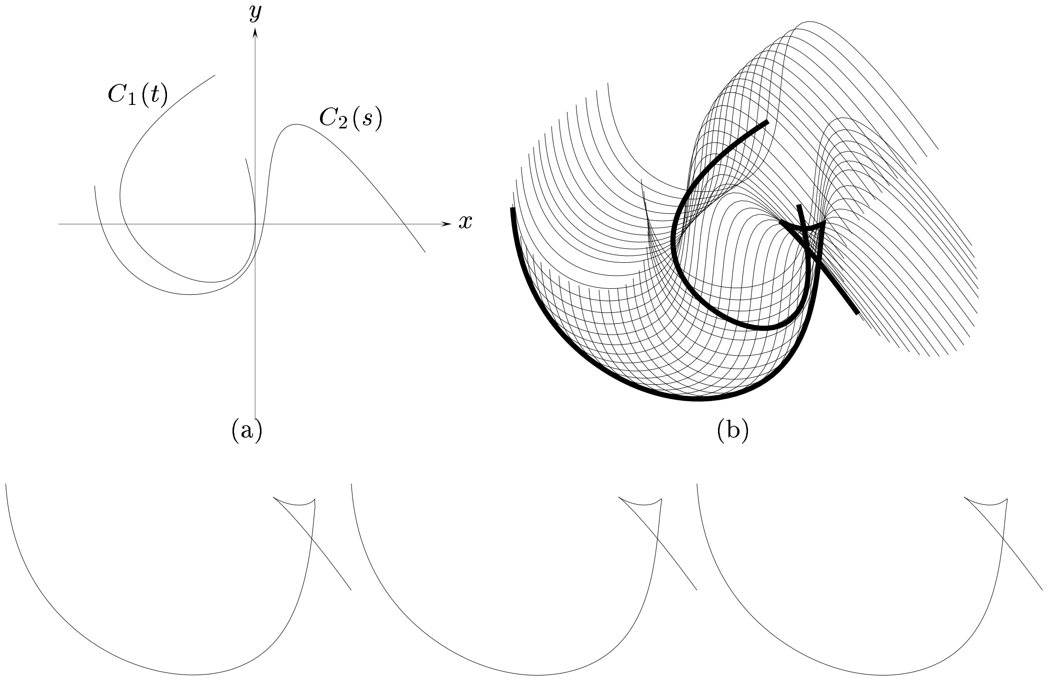

| QAC | Degree | Number of control points | | |
|---|---|---|---|---|
| | | $\epsilon = 0.1$ | $\epsilon = 0.01$ | $\epsilon = 0.001$ |
| $C_1(t) + Q_2(s(t))$ | rational 7 | 211 | 435 | 1197 |
| $C_2(s) + Q_1(t(s))$ | rational 11 | 374 | 1058 | 3158 |
| $\bar{Q}_1(t) + Q_2(s(t))$ | rational 4 | 189 | 464 | 1368 |
| $\bar{Q}_2(s) + Q_1(t(s))$ | rational 4 | 159 | 472 | 1452 |

**Fig. 3.4.** Various QAC approximations computed from cubic $C_1(t)$ and rational cubic $C_2(s)$

Figure 3.5 shows TMC approximations computed for two cubic B–spline curves. The figure compares the number of control points that are generated for different tolerances of approximation error. $\Delta$ is the tolerance of normal deviation when we approximate $s(t)$. Figure 3.5–(b) verifies the convolution curve by sweeping $C_2(s)$ along $C_1(t)$.

## 4. Conclusion

In this paper, we have presented new methods for approximating the offset and convolution of planar curves. These methods are based on two approaches: *quadratic curve approximation* and *tangent field matching*. The approximation of offset and convolution curves are computed as polynomial or rational curves within the tolerance of approximation error $\epsilon > 0$. The concepts of quadratic curve approximation and tangent field matching can be used for other geometric operations which are closely related to the normal and/or tangent directions of input curves.




(c) $\Delta = 0.1$ (degree) Number of Pts. = 45
(d) $\Delta = 0.01$ (degree) Number of Pts. = 90
(e) $\Delta = 0.001$ (degree) Number of Pts. = 99

**Fig. 3.5.** TMC approximation of two cubic B–spline curves

The 3D extension of these approaches remains an important problem for future research. Bajaj and Kim [2, 4] showed that the 3D offset and convolution of algebraic surfaces are also algebraic; however, their algebraic degrees are very high. Therefore, for their applications in practice, it is very important to approximate the 3D offset and convolution surfaces by polynomial and rational surfaces. We believe considerable research effort in CAGD should be devoted to the 3D extension of our current work reported in this paper.

## References

1. Ahn, J.-W., Kim, M.-S., and Lim, S.-B. (1993): Approximate general sweep boundary of a 2D curved object. *CVGIP: Graphical Models and Image Processing*, **55**(2):98–128.
2. Bajaj, C and Kim, M.-S. (1988): Generation of configuration space obstacles: The case of a moving sphere. *IEEE J. of Robotics and Automation*, **4**(1):94–99.
3. Bajaj, C and Kim, M.-S. (1989): Generation of configuration space obstacles: The case of moving algebraic curves. *Algorithmica*, **4**(2):157–172.
4. Bajaj, C. and Kim, M.-S. (1990): Generation of configuration space obstacles: The case of moving algebraic surfaces. *The Int' J. of Robotics Research*, **9**(1):92–112.

5. Cobb, B. (1984): *Design of Sculptured Surface Using The B–spline Representation.* PhD thesis, University of Utah, Computer Science Department.
6. Cohen, S., Elber, G., and Bar-Yehuda, R. (1995): Matching of freeform curves. *in preparation.*
7. Coquillart, S. (1987): Computing offset of B–spline curves. *Computer-Aided Design*, **19**(6):305–309.
8. Elber, G. and Cohen, E. (1991): Error bounded variable distance offset operator for free form curves and surfaces. *Int' J. of Computational Geometry and Applications*, **1**(1):67–78.
9. Elber, G. (1992): *Free Form Surface Analysis Using A Hybrid of Symbolic and Numerical Computation.* PhD thesis, Department of Computer Science, The University of Utah.
10. Elber, G., Lee, I.-K., Kim, M.-S. (1996): Qualitative and quantitative comparisons of offset curve approximation methods. to appear in *IEEE Computer Graphics & Applications.*
11. Farouki, R.T., and Rajan, V.T. (1988): Algorithms for polynomials in Bernsten form. *Computer Aided Geometric Design*, **5**:1–26.
12. Farouki, R.T., and Neff, C.A. (1990): Algebraic properties of plane offset curves. *Computer Aided Geometric Design*, **7**:101–127.
13. Farouki, R.T., and Sakkalis, T. (1990): Pythagorean-hodographs. *IBM J. of Research and Development*, **34**:736–752.
14. Ghosh, P., and Mudur, S.P. (1984): The brush-trajectory approach to figure specification: Some algebraic-solutions. *ACM Trans. on Graphics*, **3**(2):110–134.
15. Ghosh, P. (1988): A mathematical model for shape description using Minkowski operators. *Computer Vision, Graphics & Image Processing*, **44**(3):239–269.
16. Ghosh, P. (1993): A unified computational framework for Minkowski operations. *Computers and Graphics*, **17**(4):357–378.
17. Guibas, L., Ramshaw, L., and Stolfi, J. (1983): A kinetic framework for computer geometry. In *Proc. of 24th Annual Symp. on Foundations of Computer Science*, pages 100–111.
18. Held, M. (1991): A geometry-based investigation of the tool path generation for zigzag pocket machining. *The Visual Computer*, **7**:296–308.
19. Hoffman, C. (1989): *Geometric and Solid Modeling.* Morgan Kaufmann.
20. Hoschek, J. (1988): Spline approximation of offset curves. *Computer Aided Geometric Design*, **5**:33–40.
21. Hoschek, J., and Wissel, N. (1988): Optimal approximation conversion of spline curves and spline approximation of offset curves. *Computer-Aided Design*, **20**(8):475–483.
22. Kaul, A., and Rossignac, J.R. (1992): Solid interpolating deformations: Construction and animation of pip. *Computers and Graphics*, **16**:107–115.
23. Kaul, A., and Farouki, R.T. (1995): Computing Minkowski sums of plane curves. *Int' J. of Computational Geometry & Applications*, 5(4):413–432.
24. Klass, R. (1983): An offset spline approximation for plane cubic splines. *Computer-Aided Design*, **15**(4):297–299.
25. Kohler, M., and Spreng, M. (1995): Fast computation of the C-space of convex 2D algebraic objects. *The Int' J. of Robotics Research*, **14**(6):590–608.
26. Lee, I.-K., and Kim, M.-S. (1992): Primitive geometric operations on planar algebraic curves with Gaussian approximation. In T. L. Kunii, editor, *Visual Computing*, pages 449–468. Springer-Verlag.
27. Lee, I.-K., Kim, M.-S., and Elber, G. (1996): Planar curve offset based on circle approximation. to appear in *Computer-Aided Design*, 1996.
28. Lozano-Pérez, T. (1983): Spatial planning : A configuration space approach. *IEEE Trans. on Computers*, **32**(2):108–120.

29. Morken, K. (1991): Best approximation of circle segments by quadratic Bézier curves. In P. J. Laurent, A. Le Méhauté, and L. L. Schumaker, editors, *Curves and Surfaces*, pages 331–336. Academic Press, Boston.
30. Pham, B. (1988): Offset approximation of uniform B–splines. *Computer-Aided Design*, **20**(8):471–474.
31. Rossignac, J.R., and Requicha, A.A.G. (1986): Offsetting operations in solid modeling. *Computer Aided Geometric Design*, **3**:129–148.
32. Tiller, W., and Hanson, E.G. (1984): Offsets of two dimensional profiles. *IEEE Computer Graphics & Applications*, **4**:36–46.